NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

Principles for the Design of
Standard Security Protocols for
Multilevel Network Communications

by

Claudia J. Kiefer

December 1989

Thesis Advisor: David K. Hsiao

Approved for public release; distribution is unlimited.

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Naval Postgraduate School 6b. OFFICE SYMBOL (If applicable) 37	7a. NAME OF MONITORING ORGANIZATION Naval Postgraduate School
6c. ADDRESS (City, State, and ZIP Code)	7b. ADDRESS (City, State, and ZIP Code) Monterey, CA 93943-5000
8a. NAME OF FUNDING/SPONSORING ORGANIZATION 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State, and ZIP Code)	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification) UNCLASSIFIED
PRINCIPLES FOR THE DESIGN OF STANDARD SECURITY PROTOCOLS FOR MULTILEVEL NETWORK COMMUNICATIONS

12. PERSONAL AUTHOR(S)
Kiefer, Claudia J.

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Master's Thesis	FROM _____ TO _____	1989 December 21	97

16. SUPPLEMENTARY NOTATION The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Communications Protocols, Security Protocols, Computer Network Protocols, Multilevel Secure Communications, Multilevel Secure Networks (Cont'd)

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

The rapid proliferation of communications and computer networks has spawned an urgent need for comparable devlopments in network security. Significant issues such as message authenticity, transmissions confidentiality and data integrity must be addressed. Unfortunately, extremely few network designs effectively deal with such complex security issues, especially those for multilevel network environments. To encourage greater advancement in this important field, standards are needed to effectively address several aspects of network security. Specifically, standard security protocols are needed to influence the direction of industry in providing multilevel secure network designs.

In this thesis, we propose three important principles that will enhance standard security protocols designs. These include the Compatibility Principle, the Inclusion Principle and the Support Principle. We describe(Cont'd)

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED
22a. NAME OF RESPONSIBLE INDIVIDUAL Professor David K. Hsiao	22b. TELEPHONE (Include Area Code) 408-6462253 22c. OFFICE SYMBOL 52Hq

DD FORM 1473, 84 MAR 83 APR edition may be used until exhausted. All other editions are obsolete

Block 18 Continued:
Multilevel Network Security

Block 19 Continued:

the concepts of these design principles and demonstrate their benefits for security protocols in multilevel secure network communications.

Approved for Public Release. Distribution Unlimited.

Principles for the Design of Standard Security Protocols for Multilevel Network Communications

by

Claudia J. Kiefer
Lieutenant, United States Navy
B.S., University of Colorado, 1984

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN INFORMATION SYSTEMS

from the

NAVAL POSTGRADUATE SCHOOL

December 1989

Author: _____
Claudia J. Kiefer

Approved by: _____
David K. Hsiao, Thesis Advisor

_____
Magdi N. Kamel, Second Reader

_____
David R. Whipple, Chairman,
Department of Administrative Sciences

ABSTRACT

The rapid proliferation of communications and computer networks has spawned an urgent need for comparable developments in network security. Significant issues such as message authenticity, transmissions confidentiality and data integrity must be addressed. Unfortunately, extremely few network designs effectively deal with such complex security issues, especially those for multilevel network environments. To encourage greater advancement in this important field, standards are needed to effectively address several aspects of network security. Specifically, standard security protocols are needed to influence the direction of industry in providing multilevel secure network designs.

In this thesis, we propose three important principles that will enhance standard security protocol designs. These include the Compatibility Principle, the Inclusion Principle and the Support Principle. We describe the concepts of these design principles and demonstrate their benefits for security protocols in multilevel secure network communications.

TABLE OF CONTENTS

I. INTRODUCTION

A. NETWORK SECURITY OVERVIEW

The continual accessions of new applications and designs for computer and communications networks are accompanied by an ever increasing need for sufficient network security. However, current developments in network security lag far behind other aspects of distributed computing technology. The terms "network security" refer to protection against any unauthorized modification, disclosure or destruction of network information, or loss of network service leading to the nonavailability of critical information.

The security issues that are raised regarding computer networks are frequently more complex than those surrounding a single-processor system. The increase in complexity primarily stems from the distributed nature of the network. A network may be comprised of any number of component computers which are linked by a communications medium for the purpose of transferring information. For example, a network may consist of multiple hosts with several, possibly dissimilar operating systems which are connected by inherently non-secure paths. Such a configuration leads to serious concerns regarding network security including data integrity, authenticity, denial of service, and data confidentiality.

Network data is vulnerable to active threats that lead to unauthorized alterations of information. To preserve the integrity of network data, countermeasures are employed to defend against unauthorized modification of messages, insertion of fraudulent messages, deletion, replay or reordering of messages. The network must ensure that information is accurately transmitted from source to destination despite external attack or internal failure. Since network communications may be subject to jamming and active wiretap threats as well as line or node failures, maintaining data integrity is no trivial endeavor.

Authenticity refers to the validity of a message or an individual. The network must protect against fraudulent transactions by verifying the correct identities of the originator and the recipient, and by establishing the validity of the message itself. Assuring the identity of a user on a remote host may be a difficult task. Oftentimes, a network host is unable to trust the authenticity of another network host, let alone the authenticity of remote users.

Denial-of-service (DOS) is the nonavailablity of communications to authorized users of the network. A DOS condition exists if information throughput falls below some predetermined minimum. DOS may result from component failure or network overload as well as from unauthorized intervention or sabatoge. The network must constantly monitor its conditions and provide contingency measures that will enhance

the reliability, survivability and provide some continuity of operations despite any casualty condition.

The confidentiality of network data is susceptible to passive wiretapping attacks which result in unauthorized disclosures of network information. By encrypting data transmissions, the network can effectively prevent the unauthorized release of message content and deter the successful analysis of communications traffic. The use of encryption raises important questions as to the granularity and distribution of encrypting/decrypting keys.

The situation becomes even more precarious when the network system must simultaneously process data at multiple sensitivity levels. In a multilevel secure network, the system must permit access to information of multiple sensitivity levels, by users with different security clearances and needs-to-know, and still prevent users from obtaining access to information for which they lack authorization. In addition to dealing with all the security issues previously addressed, a multilevel secure network must enforce separations between processes and data of different sensitivity levels. To do so, the network must ensure data is properly labeled and clearance checks are performed before releasing any sensitive information.

B. A NEED FOR STANDARDS

The need for multilevel secure systems that can be sufficiently trusted to securely and effectively process sensitive information, is widespread in defense related environs. However, only a few such systems have been proven secure, and these were modeled as single-state machines [Ref. 1, 2]. In the past, the DOD has responded to requirements for multilevel secure systems on a case-by-case basis. However, because of the need for connectivity and interoperability between heterogeneous computing systems, this approach has become increasingly inadequate and cost prohibitive.

To encourage widespread industrial development and marketing of secure computing systems, the DOD published the Trusted Computer System Evaluation Criteria (TCSEC), and sponsored the National Computer Security Center's (NCSC) work, referred to as the Trusted Network Interpretation (TNI) [Ref. 3, 4]. The TCSEC and the TNI provide technical guidance for the evaluation of security in single processor systems and computer networks, respectively. These documents represent the culmination of DOD standards for secure computing systems.

Also recognizing the need for network security standards, the International Standards Organization (ISO) drafted a Security Addendum to the Open Systems Interconnection (OSI) model [Ref. 5]. While this document defines a number of

security services, it stops short of specifying any standard protocol designs.

There is an urgent need for additional DOD standards dealing with several aspects of network security. More specifically, standard security services must be agreed upon, and standard security protocols must be developed in order to influence industrial designs for secure computer networks.

C. OBJECTIVES AND SCOPE

In this thesis, we are concerned with certain principles for the design of standard security protocols. We propose three design principles which we consider essential in specifying standard security protocols for multilevel secure network communications. These principles include (1) the Compatibility Principle, (2) the Inclusion Principle, and (3) the Support Principle.

The first of these, the Principle of Compatibility, contends that security protocols must be designed to function cooperatively within the existing network architecture. In order to uphold network security requirements, these protocols must be compatible with the structure of conventional protocols.

The Principle of Inclusion represents a uniquely effective approach to protecting data and processes of multiple sensitivity levels. The Inclusion principle states that higher-layer protocols designed to protect at a certain

sensitivity level, must properly include lower-layer protocols which protect at the same sensitivity level.

The Support Principle is concerned with the implementation alternatives for security protocols. It suggests how security protocols may be realized within the hardware and software of the network.

D. ORGANIZATION OF STUDY

In this thesis, our purpose is to further the development effort toward standard security protocols for multilevel secure networks. Currently, standards address only the evaluation process for network security. These standards are reflected in the TCSEC and TNI.

In Chapter II, TCSEC and TNI are discussed in some detail. Here, we gain an appreciation for the intricate requirements for computer and network security. Additionally, we further our understanding of the types of services that security protocols must provide.

In Chapter III, we develop the Principle of Compatibility. We present the layered architecture of conventional protocols and suggest an appropriate framework for security protocols.

Chapter IV is devoted to the Inclusion Principle. We introduce the properties of Inclusion and note their implications for security protocol designs. Implementation requirements are fully justified.

Chapter V is dedicated to the Support Principle. Hardware and software protection mechanisms are surveyed to identify ways in which they can support network security. Numerous examples of security mechanisms and their actual implementations are presented.

In Chapter VI, we conclude by summarizing the implications of these design principles for security protocols of multilevel secure network communications.

II. CRITERIA FOR TRUSTED SYSTEMS

A. STANDARDS FOR COMPUTER AND NETWORK SECURITY

Relatively few standards exist for use in defining computer and network security. Among these, Trusted Computer System Evaluation Criteria (TCSEC) is one of the most widely acclaimed documents for security in computing. Published by the DOD in 1983 (and revised in 1985), TCSEC provides an authoritative guideline for evaluating the security features of general-purpose computer systems.

Although TCSEC was designed to be application-independent, it was recognized early-on that security requirements, as specified in the criteria, would have to be adapted or expanded in order to apply them to networked-systems. After much examination and discussion, the National Computer Security Center (NCSC) drafted the Trusted Network Interpretation of the Trusted Computer System Evaluation Criteria (TNI). The TNI was issued in 1987 as a direct interpretation for computer networks, of the general requirements set forth in TCSEC.

Together, the TCSEC and the TNI served as the basis for this thesis research. Understanding the concepts presented in these two publications is of fundamental importance in the design of secure network protocols. In this chapter, each document will be separately reviewed and its most pertinent aspects will be highlighted.

B. TRUSTED COMPUTER SYSTEM EVALUATION CRITERIA

The TCSEC was developed to serve three intended purposes [Ref. 3:p. 2]:

- to provide guidance to manufacturers of commercial ADP systems as to what security features to include in their systems design in order to satisfy the trust requirements of sensitive applications,
- to provide DOD components with a means of measuring the degree of trust that can be placed in computer systems used to process classified or other sensitive information,
- to provide a basis for specifying security requirements in acquisition specifications.

The criteria specifies a "secure" computing system as one that will *control* access to information, such that only properly authorized individuals, or processes operating on their behalf, will have access to read, write, create or delete information. From this basic definition, six fundamental computer security requirements are derived. The first four of these requirements discuss what needs to be provided to control access to information; the last two requirements address how to obtain credible assurance that access control is satisfactorily provided by a trusted computer system. A brief overview of each requirement follows [Ref. 3:pp. 3-4, Ref. 6:pp. 282-283]:

- Security Policy. An explicit and well-defined security policy must be enforced by the system.

- Marking. Each object must be associated with a "label" that indicates its security level. The label must be available for comparison each time access to the object is requested.

- Identification. Every subject must be uniquely and assuredly identified. Such identification is necessary in order to mediate each access request for information.

- Accountability. Audit information must be securely maintained so that actions affecting security can be traced to the responsible entity.

- Assurance. The computer system must contain hardware and software mechanisms that can be independently evaluated to provide sufficient assurance that the system enforces these security requirements.

- Continuous Protection. The mechanisms that enforce these security requirements must be protected against tampering and/or unauthorized change.

There are four hierarchical divisions of criteria, D, C, B, and A, where A represents the most comprehensive degree of security. Additionally, divisions (excluding D) are subdivided into hierarchical classes, C1, C2, B1, B2, B3, A1. Division C and lower classes of division B are characterized by the security mechanisms they possess and the assurance that can be gained primarily through formal testing. Systems of the higher classes in division B and division A derive their security assurances through a more rigorous analysis of the design process. [Ref. 3:pp. 5]

The following is a brief abstract of each criteria class [Ref. 3:pp. 93-94, Ref. 7:p. 101]:

- Class D: Minimal protection. This class consists of those systems that have been evaluated but that fail to meet the requirements for a higher class.

- Class C1: Discretionary Security protection. The Trusted Computing Base (TCB) satisfies discretionary security requirements through the separation of users and data. The C1 environment is expected to be one of cooperating users processing data at the same level of sensitivity. (The TCB consists of all the security-relevant portions of a system.)

- Class C2: Controlled Access protection. Systems enforce a more finely grained discretionary access control. Users are held individually accountable through login procedures, security-related auditing and resource isolation.

- Class B1: Labeled Security protection. All C2 features are required. Additionally, an informal statement of the security policy, data labeling and mandatory access control are needed. All exported information must be correctly labeled, and any flaws identified through testing must be removed.

- Class B2: Structured protection. The TCB is based on a formal security policy model which incorporates discretionary and mandatory access control. The TCB must be structured into protection-critical and non-protection-critical elements that enable it to be more thoroughly tested. Authentication mechanisms and stronger configuration management controls are

imposed. These systems are considered "relatively resistant" to penetration.

- Class B3: Security Domains. The TCB must satisfy the requirements of a reference monitor. It must be tamper-resistant and small enough to be analyzed and tested. Audit mechanisms are expanded, recovery procedures are required, and a security administrator is supported. These systems are considered "highly resistant" to penetration.

- Class A1: Verified Design. These systems are functionally equivalent to those in Class B3. However, these systems require a formal model of the security policy and a formal top-level specification of the design. Formal verification techniques result in a high degree of assurance that the TCB is correctly implemented.

The TCSEC sets forth evaluation criteria for general-purpose computer systems. However, the criteria fails to address a number of issues peculiar to network security, such as protection against data compromise and denial of service, and integrity of transmitted data. After much debate, TNI was drafted to address these outstanding issues and to provide more specialized guidance for trusted computer networks.

C. TRUSTED NETWORK INTERPRETATION OF THE TCSEC

The TNI was written to serve the same functions for networked systems that TCSEC performs for general purpose computers. Essentially, it extends the evaluation classes and

criteria to trusted network systems and components. The document is divided into two parts. Part I provides interpretations of TCSEC security features and assurance requirements. Its evaluation system is identical to that for TCSEC [Ref. 8:p. 3].

Part II describes additional security services (eg., communications integrity, denial of service, transmission security) that are of significant concern in the network environment [Ref. 4:p. IX].

The TNI provides two alternative network views for accreditation and evaluation purposes: (1) as a single unified system referred to as a "single trusted system", or (2) as a collection of two or more interconnected, independently-accredited Automated Information Systems (AISs) [Ref. 4:p. XIII].

From this first perspective, a network is regarded as an instance of a single trusted system [Ref. 8:p. 1]. It has a single TCB, referred to as a Network Trusted Computing Base (NTCB), which is partitioned among the network components [Ref. 4:p. XIV]. Collectively, the NTCB components enforce a well-defined network security policy, despite vulnerable communications paths and asynchronous operations.

The network must possess a coherent security architecture and design that correctly and unambiguously specify all security-related interfaces and services. A reference monitor must be implemented to mediate all access requests of subjects

to objects. Examples of "single trusted systems" include packet switched communications networks, end-to-end encryption systems and local area networks [Ref. 4:p. XV].

As a single trusted system, the network is evaluated using the requirements of TCSEC as interpreted for the network environment. In order to be accredited within a given class, each requirement for that class must be satisfied by the network as a whole [Ref. 4:p. XVII]. The resulting network certification is a technical statement about the strength (in terms of security) of the system, regardless of its environment [Ref. 8:p. 1].

Within each evaluation class, requirements are specified in terms of security features. For example, to be accredited at the B3 level, the network must demonstrate the following policy features [Ref. 4:pp. 90-124]: discretionary access control, protection against object reuse, data labeling, mandatory access control, identification and authentication, and auditing.

The alternative evaluation procedure is to view a trusted network as a collection of trusted components or AISs. Using this technique, each component or AIS is individually rated and certified to process sensitive information at a single level, or over a range of levels simultaneously. Then, elaborate component connection and interconnection rules are provided to ensure that the resulting network in no way violates the

mandatory security policy. This view does not offer the same formal assurances as are achieved in a single trusted system.

As previously mentioned, Part II of TNI describes a number of additional security services that are not reflected in Part I; nor do they effect the accreditation class of any network. However, each security service contained in Part II, is potentially significant given a particular network environment. Therefore, a trusted network is assigned three qualitative ratings to reflect how well it performs each service. The evaluation criteria for each service includes functionality, strength of mechanism and assurance. Table I lists these network security services, the criteria and an evaluation range for each criterion [Ref. 4:pp. 163-192].

Thus, TNI provides technical guidance for the evaluation and specification of security control in computer networks. It focuses on policy and assurance features necessary to achieve certain levels of accreditation.

Of the security requirements and services addressed in the TNI, many are implemented with the help of security protocols. For this reason, security protocols must be free of design and implementation deficiencies which can effect the services rendered. The TNI describes certain techniques, such as formal verification and testing which can assure correctness of the protocol design and operation. What remains outstanding are standard security protocols of verified designs, that will simplify the evaluation and certification process, and provide

the necessary security and assurance requirements for various network environments.

TABLE I

TNI PART II, NETWORK SECURITY SERVICES

Service Title	Criterion	Evaluation Range
Authentication	Functionality	None, present
	Strength	None to good
	Assurance	None to good
Communications Field Integrity	Functionality	None to good
	Strength	None to good
	Assurance	None to good
Non-repudiation	Functionality	None, present
	Strength	None to good
	Assurance	None to good
Continuity of Operations	Functionality	None to good
	Strength	None to good
	Assurance	None to good
Protocol-based DOS Protection	Functionality	None to good
	Strength	None to good
	Assurance	None to good
Network Management	Functionality	None to good
	Strength	None to good
	Assurance	None to good
Data Confidentiality	Functionality	None to good
	Strength	Sensitivity Level
	Assurance	None to good
Traffic Confidentiality	Functionality	None to good
	Strength	Sensitivity Level
	Assurance	None to good
Selective Routing	Functionality	None, present
	Strength	None to good
	Assurance	None to good

III. COMPATIBILITY PRINCIPLE

A. PROTOCOL FUNCTIONS AND DESIGNS

Protocols perform the functions necessary for successful communications among separate entities in computer networks. They resolve the complexities of different data formats and exchange conventions, and provide common, well-defined interfaces among communicating processes.

A multitude of tasks must be accomplished to provide for the efficient and reliable transfer of information between two networked systems. Defining a single protocol to perform these tasks would be an extremely complex endeavor. Instead, several protocols exist within a computer network architecture, to operate cooperatively, in order to carry out the communications function.

The Compatibility Principle states that security protocols must be designed to function cooperatively within the existing network architecture. Security protocols must be compatible with the structure of conventional protocols and uphold the security requirements.

In this chapter, the layered architecture of network protocols will be presented. Then, the two most prominent protocol reference models will be briefly outlined, and an appropriate framework for the design of security protocols will be recommended. Finally, a number of design objectives

for the successful development of security protocols will be considered.

B. A LAYERED ARCHITECTURE

The contemporary approach to expressing a network architecture applies a layered design technique. The protocol tasks necessary for successful communications are partitioned into several "horizontal" layers which form a functional hierarchy. Each layer performs a subset of the functions required to communicate with another system of the same layer [Ref. 9:p. 3].

Consider for example, the transfer of a file via a computer network, from its place of storage in one computer, to a user of another computer system. The procedure involves several separate tasks which can be distributed throughout many distinct protocol layers. First, a communications path must be established between the two computers. Then, some negotiation of data formats and transfer rates may be required. Eventually, the actual file transfer will occur, accompanied by procedural requirements such as acknowledgment, flow control and error detection. Finally, the connection must be smoothly terminated.

In a communications task such as just described, several relatively independent protocol layers may be employed. In the highest layer, a protocol interface may be necessary between the application process and the computer.

Intermediate-layer protocols must establish and maintain the communications path, and control the flow of data between the two host computers. Lower-layer protocols are responsible for error checking and routing data packets to their proper destinations.

In order to ensure the efficient and reliable transfer of information, an upper-layer protocol may call upon the functions of the next lower layer. In turn, the intermediate-layer protocol may engage the services of the lowest- layer protocol. Thus, a layered architecture helps distribute an extremely complex communications problem among a number of more well-defined and manageable tasks.

Another important benefit is realized from this layered design; the modularity of protocol layering enhances flexibility in network communications. It may still be unrealistic to assume that one protocol may be easily substituted for another within a particular layer. However, changes to any existing protocol can be made with a lesser affect upon the functionality of those above it.

C. PROTOCOL REFERENCE MODELS

A protocol suite is a structured set of protocols that executes the network communications function. Two independent protocol suites were designed as models for the description and specification of network architectures. The first of these protocol reference models (PRMs) was developed

by DOD; the second PRM was sponsored by the International Standards Organization (ISO). In this section, an overview of each model will be presented, highlighting the motivation for, and underlying framework of each network protocol architecture. Additionally, an appropriate reference model for the development of security protocols will be suggested.

1. The DOD Protocol Reference Model

DOD was first to develop a standard suite of communications protocols. Its motivation for a military standard protocol is revealed in two major trends relating to computer communications within DOD [Ref. 10:p. 2]:

- The rapid proliferation of computerized military devices, and the need to integrate equipment of multiple vendors.
- The wide distribution of data communications networks, especially local-area networks.

These trends generated the requirement for a common set of protocols that would satisfactorily support communications in heterogeneous network environments.

At the time the DOD protocol suite was promulgated, competing vendors were promoting their own, proprietary solutions for communication between computers. Stallings identified a number of advantages resulting from the issuance of DOD standards [Ref 10:p. 3]:

- Interoperability: By mandating the use of a common set of protocols on all DOD equipment, interoperability was achieved.

- Vendor productivity and efficiency: Since protocol conversion capabilities were no longer a concern, vendors wishing to support DOD could concentrate on developing standard protocols.

- Competition: The existence of standards theoretically encouraged competition by fostering interoperability of equipment from various manufacturers.

- Procurement simplification: Procurement deliberations were no longer required to consider protocol-conversion costs.

The DOD protocol reference model is based on a layered architecture. The complex communications task is divided into separate functions which are performed by the various network entities. Four relatively independent layers are identified [Ref. 10:p. 5]:

- Network Access Layer: As the lowest layer, its function is to provide for the exchange of data between a host computer and the network to which it is attached. Protocols within this layer must be capable of controlling access to, and routing data between various devices of the same network.

- Internet Layer: This layer is responsible for communications between entities of two or more different networks. Internet protocols must be implemented in gateways (processors between independent networks) as well as in network host computers. These protocols are responsible for

routing data among hosts and processes, across multiple networks.

- Host-to-host Layer: The reliability function is concentrated in this layer for both internet and intranet data exchange. Protocols of the host-to-host layer are responsible for error checking and proper sequencing of transferred data.

- Process Layer: This layer encompasses those protocols needed to support specific applications. For each individual application, such as file transfer or electronic mail, a separate protocol is needed to perform the communications function.

DOD standard protocols have been developed for the three upper layers of the PRM. The functions of the network access layer are performed by international standard protocols.

In order to accomplish a reliable information exchange, protocols of a particular layer must occasionally interact with immediately adjacent layers. For example, a protocol of the internet layer may utilize the services of the network access layer. However, protocols are not restricted to this procedure; application-specific protocols of the process layer may be designed to interface directly with the protocols of any one of the lower layers.

The DOD PRM was promulgated in response to an immediate need for interoperability and communications

between multiple-vendor computer networks. Coincident to the development of a DOD protocol model, a similar, yet independent effort was underway by the International Standards Organization (ISO). This organization continues to sponsor research and development of protocols to be included within an international PRM. However, DOD was unable to wait for these international standards to evolve and stabilize. In the next sections, we describe the international standards model and then provide a brief comparison of the ISO and DOD communications architectures.

2. The ISO Protocol Reference Model

The ISO-sponsored protocol model was developed in order to define standards of communications between heterogeneous computer systems. This protocol design framework is referred to as the Open Systems Interconnection (OSI) protocol reference model. The term OSI is intended to imply the mutual recognition and support of standard services and protocols by distinctly different computer systems.

One of the most fundamental issues for the ISO protocol subcommittee involved the layering of protocol functions. In order to successfully distribute the communications task, the committee needed to determine the appropriate number of protocol layers, and the services to be performed within each layer. Too many layers would unnecessarily complicate the engineering process of describing and integrating several protocol layers. Too few

layers would mean protocols would have to perform a wide range of functions, which would have imposed undue complexities in their designs. Ultimately, the ISO subcommittee agreed upon seven functional layers which serve as the basis for the OSI architecture. These OSI layers are briefly described as follows [Ref. 9, 11, 12]:

- Physical Layer: This is the lowest of the seven layers. The protocol of the Physical layer is concerned with the transmission of the raw-bit stream. It specifies the electrical representations of 1s and 0s, and details the procedure for opening, closing and maintaining the physical connection. The Physical layer protocol supports a full duplex, half duplex or simplex connection, and provides a multiplexing function for multiple data links over a single physical connection.

- Data Link Layer: This layer is responsible for converting an unreliable transmission channel into a reliable communications path. Its principle services include managing communications between directly-connected systems and providing low-level error detection. The Data Link protocol breaks up the raw-bit stream into frames and applies a checksum to each frame, in order to detect transmission errors. It also guarantees that data is correctly transmitted and received by repeatedly transmitting each frame until its receipt is properly acknowledged.

Additionally, the Data Link protocol provides flow control of data frames, so receiving buffers will not be overwhelmed.

- Network Layer: The Network layer protocol performs the routing and relay functions in point-to-point networks. It is responsible for establishing, maintaining and terminating connections between transport entities. Among its many services, the Network layer provides both normal and expedited data transfer, error detection, flow control and data sequencing. It can also provide multiplexing of two or more connections over a single network data link.

- Transport Layer: This layer is responsible for the reliable exchange of data between processes of distinctly independent systems. It ensures that data units are delivered in the proper sequence, without error, loss, or duplication. Additionally, the Transport layer protocol optimizes the use of network resources and guarantees a particular level of service quality for the upper layers. All details of the Transport service are effectively hidden from the communicating application processes.

In order to provide a wide range of functions, there are currently five classes of Transport protocols. Among the possible services offered, the Transport protocol may provide connection establishment and termination, error recovery, multiplexing, flow control and error detection. For any particular connection, the Transport protocol requirements will largely depend upon the reliability of the network

layer, and can be negotiated during the establishment of communications.

- Session Layer: The Session layer protocol provides a means for two processes to establish and maintain a connection for a definite period of communications, called a session. The Session layer protocol may support a duplex, half duplex or simplex dialogue. Additionally, it may employ a checkpoint mechanism, such that in case of transmission failure, data retransmission will commence from the last checkpoint.

- Presentation Layer: This layer is concerned with providing an acceptable syntax for the exchange of data between applications. The Presentation layer protocol resolves differences in data formats and representations. It allows communicating processes to select an agreeable syntax from a variety of data formats. It may also perform a data transformation function, such as text compression or encryption. One protocol of the presentation layer is referred to as the Virtual Terminal Protocol (VTP). The VTP converts specific terminal characteristics used by applications programs, to generic or virtual terminal features which are capable of direct computer interface.

- Application Layer: This layer provides services directly to the application processes. It is the only interface between the user and the OSI environment. While the content of the Application layer remains at the discretion of the

Application process, a few widely-used application-layer protocols do exist, such as the file transfer protocol and the electronic mail protocol.

3. An Appropriate Protocol Reference Model

The OSI model provides direct interaction between two peer entities at the physical layer. At all other layers, each entity communicates not with its peer in another host, but with its own-host entities in layers directly above and below it. Each entity invokes the functions of the next lower layer in order to perform a service for the next higher layer. Thus every communication must undergo seven layers of processing. However, where efficiency is a concern, virtually null layers may be implemented in order to streamline communications.

Still, this aspect of the OSI architecture remains quite different from operational procedures in the military model. In the DOD-sponsored environment, the use of each individual layer is optional. An upper layer entity may directly invoke the services of any one of the lower protocol layers.

Figure 3.1 provides a comparison between the DOD and OSI PRMs. ISO has acceptable protocol standards at each of the seven OSI layers and work remains in progress for the development of additional protocols, primarily in the upper layers. In contrast, DOD has issued standard protocol designs only for the upper three layers of the DOD PRM.

Entities at the network access layer make use of OSI standard protocols.

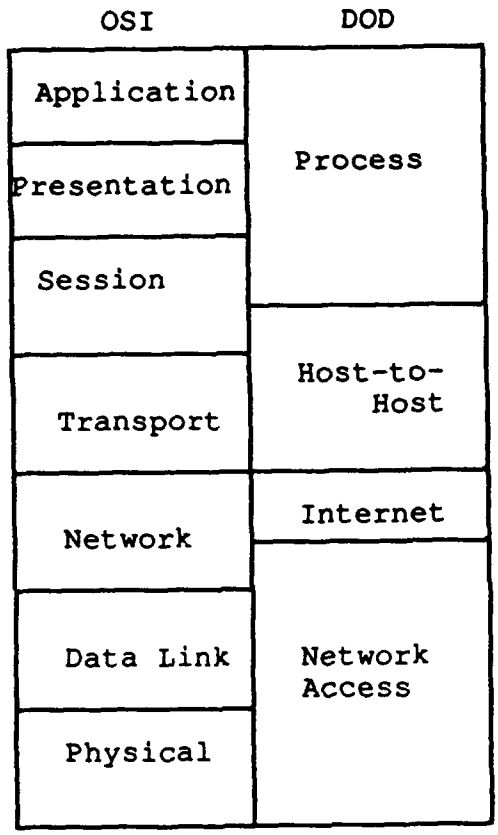

Figure 3.1. A Comparison of the OSI and the DOD Communications Architectures.

These as well as other differences exist between the DOD and OSI protocol reference models. As a result, systems supported by one set of protocols are incompatible with those supported by the other protocol suite.

Because of the nearly universal acceptance and use of the OSI PRM, vendors wishing to support DOD must incur an

additional implementation burden. Furthermore, while international standards continue to expand and improve, DOD standards are relatively static. For these reasons, DOD has declared its intention to gradually shift from its own protocol designs to international standards.

It should be relatively clear from this discussion which of the two protocol reference models would most likely be appropriate for the development of security protocols. The OSI model has received a nearly world-wide acceptance and is continuously evolving to provide new and more sophisticated functions and services. Its primary purpose is to provide a common basis for the coordination of standards development, to facilitate systems interconnection. It is within the OSI model, that security protocols should be designed to function.

D. DESIGN OBJECTIVES FOR COMPATIBILITY

The OSI reference model provides a solid framework in which standard protocols may be developed. Currently, a number of ISO standards exist at all levels of the OSI model. Most recently, ISO has issued standards for an internetworking protocol, a transport protocol and a session layer protocol [Ref. 11]. As the ISO work continues, the need for compatibility becomes increasingly important in the development of security protocols. In pursuit of

compatibility, a number of design objectives may be applied to the security-protocol development:

The service provided by a security protocol should not unnecessarily duplicate the functions of those already established [Ref. 13:p. 17]. A duplication of function only serves to increase the overhead incurred in protocol processing. If the proper OSI layer is chosen, the security protocol can utilize the services of existing protocols [Ref. 14:p. 11].

The security protocol should specify the layer in which the security service will be implemented. Otherwise, the security service would have to be implemented in different layers of separate systems, resulting in incompatibility between the two systems [Ref. 14:p. 11].

Security protocols should avoid violating the layer independence that is fundamental to the OSI reference model [Ref. 13:p. 17]. Standards developed within this framework enjoy the advantages of modularity and evolvability. Changes or enhancements can be made to the services of any particular layer without significantly disrupting the protocols of other layers. In order for specifications to be maintained abreast of current network technology, security protocols must preserve the independent nature of OSI layers.

Protocol designs should avoid all unnecessary complexities in order to facilitate the process of verification. Boundaries and interfaces must be clearly

defined, and the trusted portions of security protocols should be minimized [Ref. 15:p. 131].

Protocol designs should seek to minimize the number of alternative ways in which to implement a security service [Ref. 13:p. 17]. This effort will help simplify the process of evaluating protocol designs and issuing standards. Additionally, it will help clarify acceptable interoperations between security protocols and other network protocols.

Considerations for these objectives will help assure security protocol designs are compatible with conventional protocols. Compatibility with existing ISO standards will simplify the evaluation process and accelerate the standardization of protocol designs. Moreover, compatible security protocols will promote the underlying philosophy of "open" systems, by facilitating the protected exchange of information across heterogeneous computer systems.

IV. THE INCLUSION PRINCIPLE

A. LAYERED PROTOCOL OPERATIONS

In the OSI hierarchy, upper layer protocols define elements of large granularity and describe single operations that accomplish many things. In contrast, lower layer protocols identify objects of smaller granularity and specify conventions to complete single taskings. The combined efforts of all protocol layers are essential to the success of network communications.

The Inclusion Principle is formulated with the use of a layered approach, for example, in the OSI reference model. The Inclusion Principle is comprised of two interrelated concepts: (1) Whatever the higher-layer operations and data aggregates, a complete set of more primitive functions and data elements must be provided at the lower-layers. In other words, intermediate and lower-layer security protocols must be included explicitly to support high-layer network security and communications. (2) These subordinate protocols must protect at the same sensitivity levels as the upper-layer operations they support. For example, a Top Secret (TS) protocol, that provides secure network communications at the TS level, must be supported by lower-layer protocols which also protect at the TS level. Thus, the Inclusion Principle maintains that in facili'ating secure communications, protocols at a high layer properly

include protocols at the lower layers. Further, their sensitivity levels are the same.

Consider Figure 4.1 in which the layers of protocols are separated by horizontal dotted lines and levels of sensitivities are separated by the vertical dotted lines. The Inclusion Principle has the partitioning effect that all the protocols are compartmentalized by levels and graduated by layers. This type of layered structure involves a subtle difference from the layered structure of the OSI hierarchy.

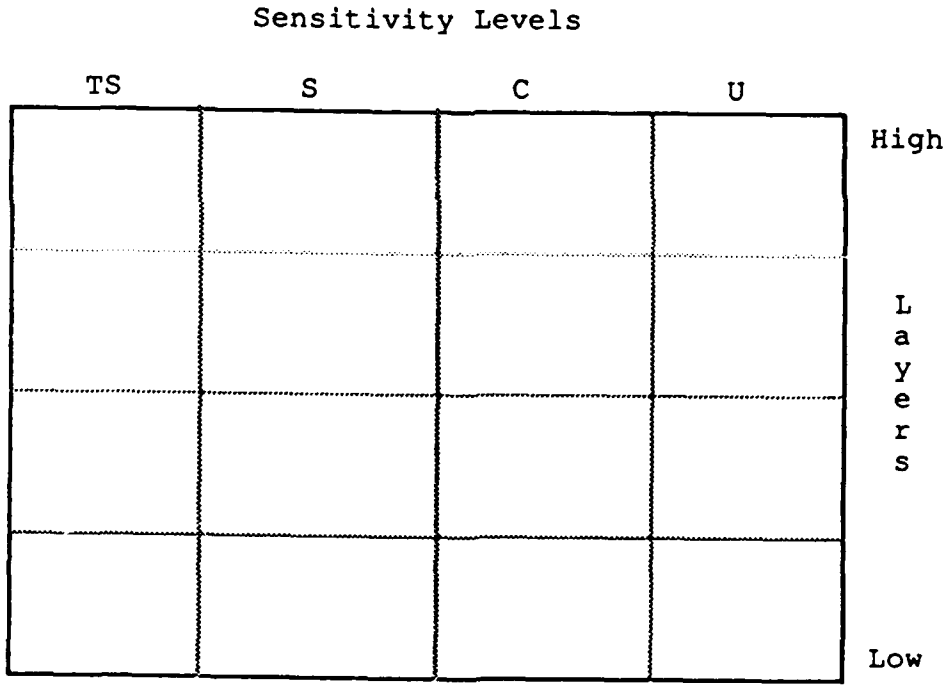

Figure 4.1. Compartmentalization and Layering of Secure Network Protocols.

Whereas, in the OSI reference model, a suite of lower-layer protocols may support some or all protocols of a higher-layer protocol suite, in our Inclusion Principle-based model, every compartment of a higher-layer protocol suite has its own suite of lower-layer protocols. In other words, one cannot use the lower-layer suite of protocols in one compartment to support a higher-layer protocol of a different compartment, since different compartments have different sensitivities for their corresponding protocols. Thus, the layer structure in our model further partitions the protocols into compartments of different sensitivity levels. This feature does not exist in the OSI reference model.

In this chapter, the Inclusion Principle will be examined and justified. Its two concepts will be separately addressed. In the next section, the need for security in all protocol layers will be demonstrated. Then, in section C, the rationale for security protocols which protect at individual sensitivity levels will be provided. Finally, in the last section, several examples will be given to compare the protocols needed to support secure network communications at different sensitivity levels.

B. A SECURITY ARCHITECTURE

In the absence of standard security protocols, a number of proprietary designs have evolved. Many of these designs focus on providing security at the OSI network or transport layers.

It is also possible to provide network security at several protocol layers. Indeed, in the Security Addendum to the OSI architecture, the ISO describes several security services and discusses where in the seven layer OSI architecture they may be placed [Ref. 13]. A summary matrix of the security services and corresponding OSI layers is provided in Table II.

TABLE II

OSI SUMMARY MATRIX OF SECURITY SERVICES AND LAYERS

	Layer						
Service	1	2	3	4	5	6	7
Peer Entity Authentication	N	N	Y	Y	N	Y	N
Access Control	N	N	Y	Y	N	N	Y
Sequence Confidentiality	Y	N	Y	Y	N	Y	N
Connectionless Confidentiality	N	N	Y	Y	N	Y	N
Selective field Confidentiality	N	N	N	N	N	Y	N
Traffic Flow Security	Y	N	N	N	N	N	Y
Connectionless Integrity	N	N	Y	Y	N	Y	N
Selective field Integrity (no recovery)	N	N	N	N	N	Y	N
Sequence Integrity (no recovery)	N	N	Y	N	N	Y	N
Sequence Integrity (recovery)	N	N	N	Y	N	N	N
Data Origin Authentication	N	N	Y	Y	N	Y	N
Non-Repudiation (origin)	N	N	N	N	N	Y	N
Non-Repudiation (delivery)	N	N	N	N	N	Y	N

Y - Yes, service shall be optionally provided.

While the OSI Security Addendum recognizes a need for security protocols, the Inclusion Principle considers the ISO recommendations inadequate for ensuring secure communications in a multilevel network environment. Security cannot be capriciously supplied at various layers of protocol reference

model. Rather, each layer of the entire protocol architecture must be properly secured in order to support multilevel secure network communications. Furthermore, as will be thoroughly justified in the next section, lower layers of protocols must be subsumed by their higher layers of protocols with the same sensitivity level, i.e., in the same compartment. For example, higher-layer protocols may be concerned with identification and authentication of communicating processes classified, say, at the secret level. But this requirement does not relieve intermediate- and lower-layer protocols from their responsibilities for preserving the integrity of transmitted data or securing the communications medium between network hosts at the secret level. In order to ensure secure and reliable network communications, security must be provided in all layers of the protocol reference model.

Certain security services are more naturally realized in the lower layers. But the ISO philosophy of placing a communications service in the lowest layer possible to achieve the desired goal, should not be misconstrued. Just because link encryption is performed at the Data Link layer, for example, does not mean that encryption need not be addressed at the higher layers. It is far more likely that additional encryption will be required at, and above, the network layer.

Further, the link encryption protocols for the top secret (TS) sensitivity level should be different from those for the secret (S) sensitivity level. Since the former supports TS

communications, the encryption algorithm should be far more complex and secure than the algorithm used to encrypt communications at the secret level. This is the application of our Inclusion Principle that is lacking in the ISO reference model.

In the ISO model, the security that can be provided by the lower-layer protocols is not sufficient for reliable and secure network communications. Consider the security requirement for individual identification and accountability across a network. This service must be provided by a high-layer protocol. Individual accountability has no significance, say, at the transport layer. Since the transport layer is host-to-host, it cannot know whether a particular user is accessing data from a particular file. [Ref. 16:p. 8]. Therefore, security must also be addressed at the upper layers.

Additionally, if a security service is provided by an upper-layer protocol, intermediate- and lower-layer protocols must ensure no disruption of that service. In other words, high-layer security must be properly preserved by the lower-layer protocol entities. An important and related concern is that of requesting and passing protocol security services between protocol layers. A full suite of security protocols (e.g., security protocols at every layer) is necessary to support secure communications. As described by the OSI model, the (n+1)-entity must be able to obtain the desired protection by invoking the security services directly within the (n)-

layer, or by requesting the service from the (n-1)-layer. In the latter case, the (n)-layer must be trusted to accurately map the (n+1)-service request to the (n-1)-layer entity [Ref. 13, p. 18]. Thus, security protocols are needed at intermediate layers of the protocol architecture, if for no other reason than to pass security-related information from one layer to another.

To summarize, a complete architecture of security protocols is needed to provide sufficient protection of network processes and data. Upper-layer protocols are needed to provide vital security services and must be adequately supported by intermediate and lower-layer entities. In turn, intermediate and lower-layer protocols are needed to uphold higher-layer security and perform additional security functions that are unattainable at higher layers. Layer entities must also act as trusted intermediaries which pass security information between layers. The Inclusion Principle requires that security protocols not only be layered, but that they also include essential supporting protocols.

C. SECURITY PROTOCOLS WITH SENSITIVITY LEVELS

The second aspect of the Inclusion Principle is concerned with the sensitivity levels or the classification of data, and the protocols which protect and manipulate that data. The Inclusion Principle maintains that a separate suite of security protocols is needed for each sensitivity level processed.

While this contention has far-reaching design implications for network security, such specialized protocols are considered essential for preserving the security of classified data.

In the DOD environment, information is classified according to the amount of damage to national security that would result from its unauthorized disclosure. For example, when improper disclosure would result in some "damage", the information is considered Confidential. When such a disclosure would cause "serious damage", the information is classified Secret. Likewise, when unauthorized disclosure of information is likely to cause "grave damage" to national security, that information is guarded as Top Secret. [Ref. 17]

In line with these definitions, the DOD requires that various levels of classified information be separately maintained. We refer to this requirement as compartmentalization. Further, the degree of protection accorded the information is directly related to its security classification. For example, confidential information may be maintained in a locked, steel filing cabinet, equipped with a General Services Administration (GSA)-approved combination lock. However, top secret information must be stored in a GSA vault or security container. Additionally, this vault must be protected by an alarm system or by a manned guard during non-working hours [Ref. 17].

From these security policy descriptions, it may be intuitively obvious that separate security protocols are needed

to process different levels of classified data. However, further justification is provided in the discussion that follows.

Underlying the Inclusion Principle is the issue of reliability. It is quite natural that only the strongest and most reliable protocols be used to protect the most highly classified data. Moreover, it is evident that some protocols are not as reliable as others. For example, consider the Unix-based TCP/IP protocols--those designed for point-to-point network communications can generally be relied upon for consistent, high-quality service. But those provided for broadcast communications could not guarantee a broadcast message would be received by all intended parties.

In this particular situation, the deficiency in broadcast communications was due to minor hardware and software flaws. However, in some cases, protocol operations may be intentionally designed without many assurance features. The ISO connectionless operation is one example. Connectionless service was designed for those contexts in which the overhead of connection establishment and maintenance is unjustified.

The point of this discussion is that reliable security protocols are of greater concern when the data to be protected is of a higher sensitivity level. Moreover, it is senseless to use highly-reinforced security protocols to protect less sensitive data.

Reliability is an expensive proposition; it is expensive to design and test reliable security protocols. Additionally, it is very costly, in terms of machine resources and processing time, to implement them. Hence, it is unwise to use the most trusted and redundant security protocols to protect data of lower sensitivity levels.

With compartmentalization, we can isolate the reliability issue from one compartment to another. The use of the virtual-machine concept may confine each compartment to a separate virtual machine, for example.

In addition to the reliability and expense considerations, data integrity is a significant concern. In a multilevel secure network, data and processes must be separately maintained by their sensitivity levels. Subjects of one classification cannot be allowed to penetrate compartments or connections reserved for other classifications.

If the network is "trusted", communications between processes are permitted only in accordance with a well-defined security policy, such as the Bell-LaPadula model. In this model, two processes can communicate only if the following Security Level (SL) conditions are enforced [Ref. 18:p. 64]:

Process A can read information from Process B only if:

SL(A) >= SL(B) (Simple Security Rule)

Process A can write information to Process B only if:

SL(A) <= SL(B) (* Property)

This restriction renders it imperative that processes communicating across a multilevel secure network share equivalent security levels.

Consider the consequences of data that is contaminated with multiple sensitivity levels. Suppose for example, because of some faulty protection mechanism, some top secret data spills into a secret compartment of the network. The access controls for that compartment are designed to protect at the secret level. Therefore, as other secret data is being properly disclosed, top secret may be inadvertently revealed.

This discrepancy is but one concern. Another consideration is the significant loss of control over top secret information in the first place. Thus, it is essential to rigidly enforce the separation of programs and data by their sensitivity levels.

The Inclusion Principle provides a natural extension of the Bell-LaPadula properties by helping to maintain the partitions between different classifications of data. Since processes must share an identical protocol in order to communicate, a separate security protocol for each sensitivity level would effectively prohibit communications between processes of different security levels.

Whether the concern is reliable communications, implementation expense or data integrity, separate security protocols for different data sensitivities provide a viable design alternative. Security protocols can be customized for

the sensitivity levels of the data they protect. In turn, the cost of using a particular security protocol will be appropriate to the level of classified data protected. Finally, these specialized security protocols will help ensure the integrity of sensitive programs and data, by enforcing partitions between various sensitivity levels.

D. SENSITIVITY LEVEL FEATURES OF SECURITY PROTOCOLS

Upholding the Inclusion Principle, we can envision many distinguishing features of security protocols which would vary considerably depending upon the sensitivity level of data to be protected. In this section, we will compare the protocols needed to support secure communications at different sensitivity levels. To illustrate, we will consider two instances of secure network communications; one at the top secret level and another at the confidential level. Many of the security services included in this discussion are drawn from those identified in Part II of the TNI.

Suppose two classified processes wish to communicate across a secure multilevel network. First, the network must ensure the authenticity of each process. If the processes desire to communicate at the top secret level, the authentication procedure might involve a complex cryptographic technique, combined with a two or three-way handshaking protocol and a time stamp. In contrast, if communication is to take place at a confidential level, each process might authenticate merely by

providing a secret password to the network. Thus, the authentication features of security protocols in the Application layer, the Presentation layer, and possibly even the Session layer would be affected by the sensitivity levels of the communicating processes.

To provide these processes with secure and reliable transmissions, the network must guarantee a certain degree of data integrity. This service would normally be provided by the Presentation layer where code and format conversions are performed. To support communications at the confidential level, the security protocol might have to detect and report any unauthorized alteration, insertion, deletion or replay of data. For top secret communications, the security protocol may in addition, have to attempt a certain amount of recovery from these random errors or unauthorized modifications. Moreover, a protocol that protects confidential transmissions would be more likely to have a higher probability of undetected errors, than one which protects top secret communications. Thus, those protocol features responsible for preserving data integrity are also altered by the data's sensitivity level.

Similarly, a security protocol that provides non-repudiation of a confidential message, would be different from one that provides this service for a top secret exchange. Non-repudiation prevents a sender from disavowing a legitimate message or the receiver from denying its receipt

[Ref. 4:p. 172]. A digital signature protocol would be incorporated in the Presentation layer to provide a non-repudiation service. As with any other security service involving encryption, the particular digital signature employed would be largely determined by the strength of the cryptographic cipher, and in turn, by the sensitivity level of protected data. Obviously, for a top secret message exchange, the strength of the encrypting algorithm would be much greater than that needed for a confidential exchange.

In order to ensure the availability of communications, the network must maintain some continuity of operations despite external attack or internal failure. To provide such assurance, Denial of Service (DOS) protocols may be implemented in the Transport or Network layers. For example, a DOS protocol could initiate a request-response message to detect the availability of a remote peer-entity. Since these protocol mechanisms increase network overhead, their use must be judiciously controlled. Therefore, to protect a confidential communications path, perhaps only one DOS protocol would be used. Whereas, to protect top secret communications, three or more such protocols might be implemented. Here, redundant security features ensure greater communications availability.

These are just some of the many ways in which security protocols are likely to change in response to requirements of different sensitivity levels. Additional protocol features which could vary with the sensitivity of data could include the

number or significance of negotiable services, acceptable error rates, acknowledgment conventions, and buffering capabilities.

In the preceding discussion, we have attempted to demonstrate the two properties of the Inclusion Principle. First, we have shown that in order to ensure acceptable network communications, upper-layer processes and data must be properly supported by intermediate and lower-layer protocols. Second, to achieve secure multilevel communications, an entire suite of protocols must be designed for each sensitivity level. Together, they introduce the concept of compartmentalization of security protocols and aggregates on the basis of sensitivity levels. This concept is orthogonal to the notion of layers. Thus, of security protocols, the Inclusion Principle dictates both horizontal layers of protocols and aggregates, and vertical compartments of sensitivity levels.

V. THE SUPPORT PRINCIPLE

A. THE IMPORTANCE OF SECURITY MECHANISMS

The TCSEC and subsequently the TNI were published to provide a means of evaluating specific security and assurance features available in "trusted" computer systems. Therefore, when seeking a certifiable design for a secure computer network, a major objective is to incorporate security mechanisms which will satisfy TCSEC assurance requirements.

Many factors combine to provide a particular level of assurance in a secure computer network. Certainly, security protocols are essential to this effort. However, a protocol is merely a set of rules designed to govern the exchange of data. In order to fulfill the TCSEC/TNI requirements, protocol designs must be properly implemented in the system's hardware and software. This then, is the essence of the Support Principle--appropriate security mechanisms must be provided within the system's hardware and software to support the proper operation of security protocols.

In this chapter, a wide variety of mechanisms that support security protocols will be considered. To facilitate this discussion, examples will be drawn from actual computer and network systems currently in use or under development. Since a comprehensive review of each system is not possible

within the scope of this thesis, only the system's most salient security features will be presented.

B. HARDWARE PROTECTION MECHANISMS

Computer hardware is used extensively in the protection of the system and user code as well as system and user data. It is instrumental in affecting various memory management techniques and aids in the control of multiple execution states. Additionally, separate hardware may be used to provide security through isolation, while also improving system performance.

1. Memory Protections

Memory management schemes are utilized to provide protection of memory-resident software and data from each other. Historically, in order to allow multiprogramming in computers, certain hardware mechanisms were designed to manage multiple processes and their data simultaneously residing in the memory. These techniques provide protection against damage or destruction to data and code, whether they were in primary or in virtual memories.

In the earliest memory protection schemes, primary memory was parceled into many separate regions, each of which had its own address space. Each resident process and its data were allocated only a certain memory region in which to operate. This restriction protected each user's memory region from being accessed by another user's process.

Several hardware techniques are used to control access to multiple memory regions. One method uses special CPU registers, called base/bound registers. The base register holds the lower address limit for the user process, while the bound register contains the upper address limit. When the CPU interprets an instruction, the address to the memory is checked against the addresses stored in these registers. The CPU ensures that the address to the memory is between the user's limits, i.e., in its region.

While this method protects itself from other user processes, it does not protect a process from inadvertently damaging its own code or data, by specifying an erroneous address within its own memory region. Moreover, considerable processing time may be required in order to reload base/bound registers with each new user process. Therefore, the degree of multiprogramming will be limited by the number of sets of base/bound registers and the amount of acceptable processing overhead [Ref. 19:p. 110].

Another memory protection technique is found in the IBM 360 series computer. This system introduced the use of "locks" and "keys" to protect main memory [Ref. 19:p. 111]. A lock in the form of an identification number is assigned to each individual block of memory. Identical numbers (locks) may be assigned to two or more memory blocks, simultaneously. Each user process must provide the matching keys to the memory blocks it needs to access.

The advantage of the lock and key mechanism lies in the ability of a user process to access multiple blocks of the memory. However, the number of individual blocks into which the memory can be partitioned is limited by the number of locks. In turn, the number of memory locks is limited to the number of bits allocated to the identification number. In the case of IBM 360, there are only 4 bits, i.e., 16 locks available [Ref. 19:p. 112].

A lock and key mechanism could offer a certain degree of support for network security protocols. Suppose for example, an individual lock mechanism is implemented to protect each security protocol. Then the user process need only present the proper keys for the protocols it needed to access. For security reasons, the maintenance and distribution of memory keys would be an important policy issue. In one design, the operating system could maintain custody of all the keys. Then, a user process would have to demonstrate a "need-to-know" in order to gain access to a particular memory key and its corresponding security protocol. In this way, the operating system could be certain to verify every access request.

The layered structure of communications protocols facilitates an alternate approach to the administration of memory keys. While the operating system might still maintain keys to all memory locks, higher level security protocols could also hold keys to lower level security protocols. In

this sense, "higher" and "lower" could mean the relative positions of these security protocols within the OSI reference model. Conversely, these terms might refer to the level of protection provided by the security protocols. In any case, higher level protocols would be permitted access to lower level protocols, without further intervention by the operating system. This method would reduce the overhead associated with processing access requests and distributing keys, but it would also lessen the security of the overall system.

An additional point is worth mentioning--because of the limited number of locks available for main memory, it is unrealistic to expect each security protocol to be individually locked. Instead, it may be more reasonable for one lock to secure all security protocols which protect at the same security level. In this way, a single key could provide access to only one level of classified information.

The limitations of these early techniques promoted the development of the memory protection mechanisms in use today. The notion of segmentation was designed to effect the requirement for nearly unlimited numbers of base/bound registers [Ref. 6:p. 205]. Here, segmentations are applied to the virtual memory which can be arbitrarily large and supported on disks by the operating system.

Segmentation is implemented in the hardware by means of a segment table. This table indicates the segment numbers

and their base and limit addresses. The operating system maintains the segment table. The CPU interprets the virtual addresses of the user programs by consulting the segment table. In this way, every access request is checked for legitimacy by the CPU.

Segmentation offers greater protection and more flexible support for network security protocols. Once again, protocols may be clustered according to their security levels. However, using segmentation, all protocols which protect at the same sensitivity level could be grouped into a single segment (rather than behind a single lock). Since virtual memory offers far more segments than real memory has locks, segmentation provides security at a finer granularity. In other words, a greater number of sensitivity levels may be defined for security protocols and the data and programs they control.

Additionally, segmentation provides a way in which two or more programs may be permitted different access rights to the same data segment. The access rights are stored as control bits in the segment table entries associated with those program segments.

To illustrate how segmentation may be used to support secure network communications, consider two network applications which desire to communicate at the "secret" level. Upon execution of these application programs, their associated segment tables (or portions thereof) are read into

the main memories of the hosts to which they are attached. The host operating system must then verify each application's request to access a virtual memory segment, by ensuring the segment name is entered in the application's segment table. If access is authorized, the operating system must also check the control bits associated with this segment table entry, to determine the type of access (read-only, execute-only, write) to be granted. In this scenario, at least one entry in each application's segment table must name the segment in which the appropriate communication protocols are stored. In order to communicate, the two applications must use an identical protocol. Furthermore, that protocol must be able to protect the communications exchange at the secret level. Thus, segmentation and virtual memory assist in providing secure network communications. Segmentation provides hardware-reinforced protection of security protocols, while these protocols secure the communications exchange between network applications.

Another memory management technique, called "paging" is used in the VMS and DEC systems. Each program is partitioned into uniformly-sized pages; memory is divided into the same-sized frames. The operating system maintains a page table with page numbers and frame addresses. Paging is designed to make more efficient use of the memory space by staging extra pages on the secondary storage. Since both program and data pages are of the same size as the memory

frame, paging allows necessary programs and data to be brought into the memory in real time [Ref. 20:pp. 244-254]. We can also assign access privilege bits to the page entries for access control of pages.

Combining these two approaches to memory management offers the unique efficiencies of paging and the inherent protection of segmentation. Paged-segmentation is used for example, by the Multics operating system and was originally implemented on a GE/Honeywell-645 machine [Ref. 6:p. 209].

To implement a paged-segmentation scheme, programs and data are divided into logical segments. Then, each segment is further partitioned into fixed-sized pages. Each code or data item is addressed with a segment name and an offset. The base address of the page table containing all the pages of the segment is maintained in the segment table. Thus, two-level address translation is needed--one to determine the segment number for a named segment, and one to determine the page and therefore the frame in which the page resides.

Support for network security protocols by a paged-segmentation scheme, would be similar to that provided by segmentation alone. Protocols which protect at the same sensitivity level could be stored in a single page-frame. The operating system would then use the two-tier addressing scheme to verify access requests for security protocols.

These memory management techniques emerged from the desire to accommodate multiprogramming. While they were primarily designed to make efficient use of system memory, such implementations are highly effective protection mechanisms.

2. Multiple Execution States

The advent of multiprogramming generated the need to protect not only system memory, but other system software. More specifically, a mechanism was needed to protect the internal state of the CPU during the execution of a process. Since the system was expected to support several processes simultaneously, multiple execution states were required.

A simple multiprogramming system may enforce only two different states, in order to separate system programs from user programs. For example, only the operating system should execute instructions to control I/O for all user's programs. In order to accommodate these "privileged" instructions, a privileged state and a user state are defined. Then, when a user program wishes to accomplish I/O, it issues a system call, and the hardware switches from user to privileged state. Finally, the operating system validates the I/O request and performs the operation accordingly.

In a two-state system, security protocols would also be considered privileged instructions. Then, in order to invoke the security protocol, the user would initiate a system call. The hardware would switch from user to

supervisor mode and the operating system would then verify the user's request. If the request was valid, the operating system would activate the security protocol on behalf of the user.

A two-state system is severely limited. Most significantly, it cannot support multiprogramming. While the two modes protect the operating system from the user processes, it does not protect the users from each other.

Most general-purpose computer systems support multiple execution states which are frequently implemented in a layered hierarchy. Each state (also called a domain) is represented as a separate layer, where the "lowest" layer is the most privileged state. A process executing in the "lowest" layer has access to all instructions of the "higher" layers. Whereas, a process executing at the "highest" layer has very limited access to the more sensitive operations of the "lower" layers [Ref. 19:p. 117].

The need to support multiple execution states or domains is firmly grounded in the TCSEC and the TNI. Even a system that is trusted to the relatively low C1 level must maintain a separate domain for the execution of TCB. This requirement is essential to ensure TCB code and data structures will not be inadvertently or maliciously damaged by untrusted subjects.

The Multics System is an example of a multiple state machine. The Multics operating system implements multiple

protection levels in concentric circles (rings) around the system's hardware. Each ring represents a distinct state. The most "trusted" processes execute from the inner-most ring and have access to all other system and user instructions.

A multiple state mechanism offers a number of favorable implications for network security. First, a multiple state machine can protect several levels of data and code simultaneously. Second, its hardware implementation can provide a strong mechanism to enforce an access control policy that all users must follow. These capabilities are important inputs to a reliable and secure network design.

A network host computer with multiple execution states may be relied upon to enforce the mandatory access control requirements of a definite security policy. If such assurance could be achieved in all network hosts, then the only remaining concern would be to secure the communications links and interfaces between these components. (Here, we knowingly disregard the issue of data integrity.)

However, a multiple state machine is not a panacea for system security. For example, it cannot enforce discretionary access controls, since all subjects operating at inner layers have unlimited access to object of the outer layers. Without the implementation of discretionary access controls, a network would be unable to attain <u>any</u> level of National Computer Security Center (NCSC) certification.

In addition, multiple execution states do not provide sufficient layering or controls to allow the concurrent processing of different levels of classified information. The multiple state machine was designed to accommodate only unclassified information. Furthermore, it was built to operate in a relatively friendly environment of multiple users. Therefore, the functional definitions of the various layers do not include provisions for multiple classification levels.

In order to achieve a multilevel secure system, many more levels of execution would be needed. At least three more levels would be required to accommodate the basic military model for classified data (e.g., confidential, secret, and top secret). Furthermore, additional levels may be desirable to allow for discretionary (need-to-know) control within each classification level. Of a separate, but related note, the gates which enforce the separations between layers must be sufficiently strong to prevent the intermingling of processes and data from one level of classification to another.

In a multilevel secure network, all network hosts must also enforce multilevel security. (Any host that could not be trusted to provide multilevel security would have to operate at a system-high level.) Although a multiple state machine is a desirable component of a secure computer network, it is not adequate to guarantee the security of

multiple classifications of information that are concurrently processed. Multiple execution states are but one of many security mechanisms needed in the components of a multilevel secure network.

3. Dedicated Processors

The proper operation of a multiple state system requires significant processing power in order to perform continuous and rapid process switching. For this reason, as well as several others, separate processors are often included in the system's design which are dedicated to performing hardware security functions. Such a processor might be a microcomputer used to manage file systems and I/O or a minicomputer used to perform complex computations for elaborate protection schemes. Additionally, a separate processor may be used to perform system monitoring and audit functions.

A current developmental effort which features the use of specialized processors is the CANEWARE program. Sponsored by the National Security Agency in contract with Motorola, INC., CANEWARE is expected to provide high-performance security services for host computers on long-haul, packet-switched networks [Ref. 21]. Its two principle devices include a CANEWARE Front End (CFE) and a CANEWARE Control Processor (CCP). CFE performs encryption for network data and enforces access control policies. CCP maintains the security database, conducts network security audit functions

and provides centralized administrative monitoring and control. CANEWARE is being designed for certification by NCSC, at the B2 level.

Dedicated processors offer additional benefits in network security through isolation. Protection is enhanced by isolating security protocols and service mechanisms from the operating system and from the users' spaces. Also, system verification is simplified by locating all security functions in a physically separate device.

The LOgical Coprocessing Kernel (LOCK) project is currently detailing the design specifications for a security-enforcing module called the system-independent, domain-enforcing, assured reference monitor (SIDEARM) [Ref. 22]. The SIDEARM design specifies a separate computer with specialized processors and its own memory that controls access to all resources of the host computer.

All SIDEARM security functions are distributed across individual processors, which may be real or virtual. For example, a single processor acts as a front-end filter to screen out illegal requests from the host. Another processor manages access to the system's resources, including primary and secondary memory. Still another processor performs all audit related functions.

SIDEARM is part of an ambitious development project to produce a generic hardware-oriented solution for

multilevel security on general-purpose computers. It is expected to meet TCSEC requirements for the A1 certification.

To summarize, hardware protection mechanisms for computer and network systems are quite extensive. Memory protection mechanisms provide for the separation of user and system processes and data in memory, thus minimizing any damage that may result in case of inadvertent error. Multiple execution states and ring structuring allow for concurrent operation of multiple processes, each with its own special privileges. Dedicated processors offer advantages in system performance, while effectively isolating security functions from the operating system and other users' spaces. In the next section, software protection mechanisms will be considered, which complement the security of hardware mechanisms in the design of multilevel secure networks.

C. SOFTWARE PROTECTION MECHANISMS

In order to ensure sufficient security of system and user data, software protection mechanisms are needed to augment and reinforce mechanisms of hardware security. For example, consider the hardware implementation of multiple execution states; privileged states run contrary to the principle of least privilege [Ref. 23:p. 207]. Since processes which execute from within the privileged state often have more privileges than required for the task, additional software mechanisms must be utilized to assure adequate security.

In this section, we will present three major types of software security: access control, isolation and encryption. Once again, examples from the literature will be used to support our assessments.

1. Access Control

An access control mechanism is used to enforce the system's security policy by mediating every subject's (e.g., program's) request to access an object (e.g., data). Subjects are usually users, programs or processes, while objects may be processes, files or other types of data. Two factors determine the effectiveness of access control mechanisms: First, each subject must be properly identified and authenticated. Second, information within the system which specifies access rights, must be protected from unauthorized modification [Ref. 23:p. 191].

An access control mechanism may be represented by a matrix in which each row identifies an individual subject and each column specifies a certain object. Entries within the access control matrix indicate the individual subject's rights, such as read, write or execute, to a specific object. The access control matrix is a dynamically changing mechanism. Matrix entries, as well as the subjects and objects to which they pertain, are likely to be altered during program execution.

The access control mechanism must monitor all accesses to objects and all commands to transfer or revoke

access rights. Additionally, the mechanism must ensure that the use of physical resources constantly reflects the logical permissions, as represented by the access control model. Otherwise, data may be exposed to unauthorized users [Ref. 23:p. 200].

A recent application of an access control mechanism is provided by the Boeing Aerospace Co. Multilevel Secure (MLS) Local Area Network (LAN) [Ref. 24]. MLS LAN was designed to meet the TCSEC A1 level of certification. MLS LAN enforces a security policy which includes discretionary access control. An access control matrix is utilized to maintain proper discretionary controls. For packet-oriented transmissions, the packet source and destination are the subjects, while each packet is an individual object. In a connection-oriented system, each connection is viewed as a separate object. Participants in the connection (subjects) are then given read and write, or read-only access to the connection (object).

The access control matrix can be implemented in many forms. In one version, an access control list, each object has a separate list which identifies all subjects that have access to the objects, as well as what that access is [Ref. 19:p.169, Ref. 6:p. 215]. The advantage of this implementation is that different users can share the same object without the need for a separate object entry in each user's directory. Moreover, the owner of the object has

complete control over who has access to the object. The owner may extend or revoke access rights at any time. The disadvantage of an access control list is that it is particularly expensive to search. If each request is checked, locating the object, then finding the user, and finally validating the type of access requested can be a very time consuming process.

Access control lists are used by the Multics system to protect files in long term storage. Each file segment has its own list, with an entry for each user. Access list entries reflect not only the type of access (read, write or execute) but also the range of Multics rings to which the user has access.

An access control matrix may also be implemented by a capabilities list. A capability is a kind of ticket giving a subject certain access rights to an object [Ref. 6:p. 218]. A subject may possess any number of capabilities. However, a capability may only be created through a user's request to the operating system.

In a capability-based scheme, a process executes within a certain domain. The domain is a collection of all current capabilities possessed by the process [Ref 19:p. 169, Ref. 6:p. 219]. As the process executes, it may call a subordinate process and pass it certain objects. Additionally, the subordinate process may have other capabilities that are not duplicated by the calling process.

As a result, the subordinate process executes in its own unique domain.

Since capabilities are used to control access rights, domains must be protected from normal users. One method is to isolate the domains in user-inaccessible memory segments. Another option is to distinguish capabilities from other objects by tagging them as privileged data. Of course, then tags must also be protected.

A significant benefit of a capability-based protection mechanism is that it eliminates the need to search for current access rights. If a process desires access, it must present the appropriate capability. However, when an access right needs to be revoked, the problem is far more complicated. The access right must first be found in capabilities which are distributed throughout the system.

Actually, in most systems, access control solutions use a combination of access lists and capabilities [Ref. 20:p. 386]. Initially, access control lists are searched to validate a subject's request for access. Once approved, a capability is attached to the process which indicates all subsequent and subordinate access rights. This approach enhances system performance by minimizing the revocation problems and improves response times for subsequent access checks.

In a secure multilevel network, there are many ways in which access control mechanisms may be utilized. For

example, in one network configuration, each host could be responsible for preserving the security of its own data and all processes originating from it. In this case, each host might maintain a capabilities list for all current processes. Among the capabilities listed for each process might be the level at which the process executes.

Additionally, each host might also maintain an access control list of all hosts attached to the network and a range of security levels at which they were trusted to operate. Then, when process A on Host 1 attempts to communicate with process B on Host 2, several access control checks would be initiated. First, Host 1 would check to see the level at which process A operates, say "secret". Host 1 then checks the see if this security level is within the range of Host 2. If so, Host 1 would send the identity and security level of process A to Host 2. Host 2 would then check to determine if process B was also operating at the secret level. If so, a connection between the two processes would be granted.

A different network arrangement might incorporate a centralized database to maintain a record of access rights of all hosts on the network. Then some network overseer would use the information contained in the access control database to control access to cryptographic connections between network hosts. Access to a connection would be granted only if the security level of the connection fell within the security range of the requesting host.

2. Isolation

The concept of isolation is as pervasive in software protection mechanisms as was evident in hardware mechanisms. In the last section, memory management was shown to be an essential hardware protection mechanism. Memory protection techniques provide a logical separation of the memory region of one user from that of another. Current operating system software takes this notion of isolation one step further, so that each user not only has its own logical memory, but logical files, logical I/O devices, and other logical resources, as well [Ref. 19:p. 171, Ref. 6:p. 264].

Such an operating system was first made generally available by IBM and was appropriately named the Virtual Machine or VM operating system. This system was designed to accommodate multiple operating systems executing on the same system's hardware.

A specialized control program operates as an interface between the system's hardware and two or more different operating systems. The control program provides the only existing interaction between system hardware and subject operating systems. Thus, the control program offers the benefits of a secondary security layer. Even if the user exploits a flaw in its own operating system, it may reach only the level of the control program [Ref. 19:p. 172, Ref. 6:p. 265].

The control program enforces the separation between different users and their operating systems, and between different users and the hardware itself. This improved level of protection is attained at the cost of an additional layer of complexity in the computer system design.

Isolation is an extremely important consideration in the design of a multilevel secure network. Its many users and components systems represent a significant threat to the integrity and security of network resources. In order to protect the system, each user must be isolated from all others, and every access must be strictly controlled. Additionally, a reliable separation must be maintained between various levels of classified information.

In a computing network, each host computer may be responsible for protecting the users and resources within its perimeter. To do so, each host must be sufficiently isolated from all other network hosts. This isolation is often provided in the form of a Trusted Network Interface (TNI).

A TNI acts in a manner similar to the control program of the VM operating system. In other words, the TNI enforces the separation between different hosts and between the hosts and the network itself. The TNI is typically a combination of software and hardware, and its security functions would normally include: host authentication, host-to-host encryption, mandatory and discretionary access control,

systems audit collection, and the maintenance of multiple security levels.

The TNI is "trusted" because it has been thoroughly tested and verified to perform as it was intended. While each component host must maintain the security and integrity of information within its own boundary, the TNI will ensure the protection of information on network connections.

Historically, operating systems have been known to possess certain inherent weaknesses which can jeopardize the security of the entire system. Such flaws can include unreliable I/O processing, ambiguous access policies, incomplete mediation and trap doors [Ref. 6:p. 276].

Persistent attempts to develop reliable and secure operating systems have been largely unsuccessful for a number of reasons. First, the operating system software is typically very complex. Second, necessary or desirable security controls are not precisely defined. Finally, it is difficult to verify the correct operation of the security controls that are in place. [Ref. 25:p. 142]

One relatively successful approach to the design of a provably secure operating system focuses on the intrinsic benefits of isolation. (Note, this is a different type of isolation than the one discussed previously). The objective is to isolate all the security mechanisms in a central locality at the very lowest level of the operating system. This nucleus or core, called the security kernel, is

responsible for performing all security functions for the entire computer system.

The security kernel offers several advantages for a secure operating system design [Ref. 6:p. 267]:

- Separation. By isolating the security mechanisms from the rest of the operating system and the users' spaces, the mechanisms are more easily protected from penetration and modification by unauthorized users.

- Unity. Since one set of code is responsible for performing all security functions, it is easier to develop and understand that code.

- Modifiability. The kernel's modular design facilitates changes to, and subsequent testing of protection mechanisms.

- Compactness. Since it performs only security functions, the kernel is likely to be relatively small.

- Verifiability. If the kernel is small, the verification process is considerably less complex than that required for an entire operating system.

- Coverage. Since all requests for access must pass through the security kernel, it can easily check each access to a protected object.

The effectiveness of the security kernel design depends largely on a few fundamental design principles. First, the security policy must be precisely defined. All access permissions should be completely described by a formal security model, so the functions that the kernel must provide

can be properly identified. Second, security mechanisms to be included in the kernel should be chosen so that collectively, they will mediate all accesses to protected objects, in accordance with the specified security policy. These security mechanisms must be thoroughly protected from unauthorized access or modification. Finally, system performance and complexity must be constantly weighed against the functionality of individual security mechanisms. While the kernel can be constructed entirely in software, considerable hardware support may be needed to achieve adequate performance [Ref. 25:p. 146].

There must always be a trade-off as to which security functions are implemented within the security kernel. It may seem desirable to include all security related functions without exception; however, this strategy would defeat one of the most important features of the security kernel design--the kernel must remain small so it can be thoroughly and properly verified. Furthermore, while the security kernel mediates all access requests to ensure they are permitted by the system's security policy, some other method must be used to legitimately access the kernel and to update the security policy.

One of the most successful implementations of the security kernel design is the Honeywell Secure Communications Processor (SCOMP). SCOMP was the first system to be certified at the A1 level by the National Computer Security

Center [Ref. 1]. While the SCOMP security kernel is developed entirely of software, it runs on a hardware-enforced ring mechanism.

Complete mediation and isolation is provided by a SCOMP Security Protection Module (SPM). SPM is constructed in hardware, and is situated between the system processor and the I/O controller and memory. In this way, SPM is able to mediate all processor requests and validate them prior to accessing memory or I/O devices. Because SPM is realized in hardware, I/O device drivers can reside outside the security kernel.

The SPM design offers significant benefits. First, the security kernel is smaller and less complex because it does not have to support so many different devices. Second, I/O capabilities can be modified without affecting the kernel or its verified design. Finally, fewer kernel calls result in reduced overhead and improved system performance.

The SCOMP security kernel administers all security mechanisms and controls all access to the system. Access control is effected in accordance with the embedded security policy. The security kernel performs all memory and resource management, process scheduling, trap and interrupt handling, and auditing. It supports objects which include segments, devices and processes, and offers an extensive range of functions to each process.

The SCOMP design contributed significantly to its successful implementation. Honeywell's strategy was to build the security kernel first, and then construct the remaining portion of the operating system around it. This approach helped to minimize the size and complexity of the security kernel and improve overall performance of the system. Additionally, SCOMP's layered design helped ease the verification process.

The Honeywell SCOMP offers several promising implications for the design of a secure multilevel network. First, its certified A1 level design could become the foundation of each network host. Admittedly, a certifiably secure multilevel network is much more than a collection of verified hosts. But incorporating trusted hosts is a positive step toward achieving a distributed trusted computing base.

Additionally, the design objectives used in the SCOMP operating system, may be directly extended to the multilevel secure network. For example, SCOMP's hardware reinforced access control mechanism provides higher performance than one implemented strictly in software. This technique may be used to achieve efficient and reliable mediation between network hosts.

One ambitious Navy implementation attests to SCOMP's wide range of potential network applications. The Navy is using a SCOMP design to allow two networks operating at

different security levels to communicate with each other [Ref. 9:p. 227].

3. Encryption

Encryption is one of the most important and widely used security mechanisms in computing networks. Encryption is the process of encoding individual letters or entire words or phrases, in order to mask the true meaning of a message [Ref. 3:p. 23]. Current encryption techniques generally require three elements: One ingredient is the plaintext; this is the message expressed in some natural language. The second input is the encryption function or algorithm. The third element is a unique key which is used in conjunction with the encryption function, to produce the ciphertext (encoded) message.

Encryption functions or algorithms range from simple substitution (cipher) schemes to elaborate mathematically-based transformations [Ref. 1:pp. 59-125]. Frequently, the encryption algorithm is a well-known public standard. Such openness allows two mutually suspicious parties (i.e., processes of different network hosts) to communicate in a cooperative manner. If both parties use the same encryption function, each participant may place greater trust in the integrity and authenticity of information received from the other party.

In other cryptographic systems, the transformation algorithm is secretly maintained within a cryptographic

device. Although staunchly protected, if the device was to fall into an intruder's hands, it would still be very resistant to penetration.

Regardless of a public or protected encryption algorithm, the key used in conventional ciphers is always kept secret. Conventional cryptosystems use a single, private key to both encrypt and decrypt the message. In this way, only one encryption function must be employed and the order of the encryption and decryption, makes no difference [Ref. 10:p. 324].

In a conventional key system, if the system has n users, then n*(n-1)/2 keys will be needed so each pair of users may share a unique key [Ref. 3:p. 89]. Furthermore, keys must be changed frequently, in order to maintain their secrecy. Under such circumstances, the safe distribution of so many keys can become extremely challenging.

Public key systems were designed to alleviate the shortcomings of conventional key systems. Public key cryptosystems employ two distinct keys per user; one is used to encrypt the plaintext, while the other is used to decrypt it. Since encryption and decryption are inverse functions, it does not matter which operation is accomplished first.

To illustrate, suppose User A is assigned two keys. One of these is openly published or distributed. The other key is kept secret by A. Any user can send an encoded message to A, by encrypting it with A's public key. The

message will remain secret until A decrypts it using his own private key. Moreover, if A wishes to send a secret message to another user, say B, A may encrypt the message using his own private key. Then, as long as B can identify that the sender of the message was A, B can decrypt the message using A's public key.

This scenario suggests a weakness in the public key system. The recipient of the message may require some assistance from the network in order to determine the originator of an encrypted message and thus choose the correct public key with which to decrypt it. Additionally, the author of a message may disavow it at any time, simply by claiming that his private key had been compromised [Ref. 11:p. 147].

Even in public key cryptosystems, key management can be a major issue. Intricate numerical techniques are needed to devise the pair-wise keys of public cryptosystems. Additionally, users' private keys must be given the same degree of protection as the data which they encrypt. Further, key distribution is frequently a problem, since all key changes must occur in unison. In other words, if a user changes his private key, then all copies of the corresponding public key must be changed simultaneously. There must be some form of central authority that will maintain responsibility for authenticating key changes and correctly distributing public keys upon request [Ref. 11:p. 147].

In computer networks, encryption may be applied to the communications links between network hosts or across the entire network, including all hosts. The first type of encryption is called link encryption and is incorporated in the OSI architecture at the second lowest (Data Link) layer.

Link encryption protects data during transmission between network hosts. However, data that is inside the host, remains in the clear. Link encryption is particularly appealing in networks where hosts are relatively secure and trustworthy. In this situation, link encryption is a highly efficient and effective means to protect data transmissions on an insecure communications medium. The most significant drawback of link encryption is that sensitive data remains vulnerable to attack while it is inside unsecure network hosts.

In contrast with link encryption, a second type of encryption may be applied from "end-to-end", across the network. End-to-end encryption (E3) may be implemented at any OSI level, above the network layer.

E3 may be used to achieve a secure communications channel between any two network processes. Messages remain encrypted from source to destination, even within intermediate hosts. Furthermore, E3 may be selectively applied to secure the transmission path for an entire communications session or for a single message exchange.

Either link encryption, end-to-end encryption or both may be offered on a single network. In any case, innumerable encryption keys are needed to implement such methods.

The basic requirement to distribute and safeguard large numbers of encryption keys has lead to the development of automated key servers. The key server is a process that issues keys as needed to network users [Ref. 3:p. 382]. Each user registers a unique key with the key server. When two parties wish to communicate, one of them requests a "session" key from the key server. The key server generates a new key and sends copies to both users. Each copy of the session key is encrypted with the appropriate user's key that is on file. Upon receipt of the encrypted session key, each user decrypts it and uses it for secure communications. At the end of this session, each user destroys the session key.

An automated key server offers a number of benefits to secure networks. First, it can ensure the authenticity of communicating parties. Additionally, it can maintain an up-to-date registrar of network users. And finally, it can efficiently support end-to-end encryption.

In summary, software protection mechanisms promote efficiency and flexibility in the design of multilevel secure networks. Access control mechanisms ensure each subject possesses only those privileges necessary to complete the task. IBM's Virtual Machine enforces the separation between users and their operating systems, thus adding a second layer

of security to the system. A security kernel implementation provides a mechanism that ensures complete mediation, yet is sufficiently simple and small to be completely verified. Encryption protects data during transmissions and helps ensure only authorized processes are allowed to communicate.

D. DESIGN CONSIDERATIONS FOR SECURITY MECHANISMS

The design of a multilevel secure network begins with a precisely defined security policy. From this policy definition, a list of network security services must be compiled. Then, in order to provide these security services, protocols are developed to regulate the communications between network processes, in accordance with the specific security policy. Finally, protection mechanisms are selected and developed to support the network security requirements.

Security mechanisms perform three basis functions: First, they enforce the rules of procedure as set forth by security protocols; second, these mechanisms actually implement the security services; and third, they precisely execute the security policy.

The preceding discussion surveyed a variety of hardware and software mechanisms which can be used to support network security. Just which protection mechanisms will be incorporated in the network requires a number of important design considerations. A systems architect must not only consider the function and strength of the protection

mechanism, but also its effect upon the system's performance. He must seek to achieve a particular level of security, but still incur only a minimal cost in processing overhead. The performance resulting from the implementation of a particular security mechanism should compare favorably with network performance without that mechanism.

In general, hardware mechanisms provide higher security/performance-cost ratios than do software mechanisms. This is one of the main reasons for the recent trend in computer network designs--to incorporate separate or front-end processors which are dedicated to performing only security-related functions.

It is inconceivable however, to implement in hardware, all the support mechanisms necessary for adequate network security. Software mechanisms offer characteristics of flexibility and evolvability that are not available in hardware. Software mechanisms facilitate the layering of access rights, and are critical to providing multilevel security on computer networks.

The selection of protection mechanisms must also depend upon the network security policy to be enforced. The TCSEC/TNI outlines the security policy for each class of certification. For example, at the B1 level, both mandatory and discretionary access controls must be enforced on the network [Ref. 12]. The Bell-LaPadula model must be implemented as the basis for all access controls, while

additional user-controls must be used for discretionary access permissions. (The Bell-LaPadula model is a formal description of authorized information flows in order to maintain data secrecy in a multilevel environment.) Additionally, accurate security levels and labels must be provided for all controlled subjects and objects. Despite these requirements, the TCSEC/TNI do not specify the security mechanisms needed to support them. Trusted computer and network security criteria can only be achieved through implementation of appropriate hardware and software protection mechanisms. Such details of the network security policy remain the joint responsibility of the sponsoring customer agency and the systems designer.

Similarly, protocol specifications do not include the mechanisms needed to support them. Security protocols describe the access rules and exchange conventions necessary to comply with the network security policy. They frequently define strict requirements for data structures and formatting. However, well-designed protocols are free of any implementation details. They specify the process necessary to complete a task, but not the mechanism by which it is accomplished.

For these reasons, appropriate protection mechanisms are significant design concerns for the system's architect. Hardware and software protection mechanisms are essential elements of secure computer networks. They are the

instruments through which a definite security policy is executed; the vehicles necessary for secure information flow. While protocols govern the behavior of the computer process, protection mechanisms provide the means to enforce desirable behavior.

VI. CONCLUSION

In this thesis, we noted a rapidly expanding demand for secure network communications. We observed numerous complex issues associated with network security, many of which can be effectively resolved through the use of secure communications protocols.

Security requirements for various network environments are well defined. However, few efforts have been truly successful in achieving verifiable designs for multilevel secure networks. In order to encourage more widespread development of multilevel secure networks, security protocols must be carefully designed and standardized. Such standards will foster substantial investments in compatible, yet secure designs for network communications.

Three principles were proposed which we believe will enhance the quality of standard security protocol designs. These include the Compatibility Principle, the Inclusion Principle and the Support Principle. Each of these principles offers a significant improvement to proprietary protocol designs.

The Compatibility Principle contends that security protocols must be designed within the layered architecture of the OSI reference model. Compatible security protocols offer several benefits. Secure protocol designs may be able to

utilize protocol functions and services already established within a particular OSI layer. This techniques would achieve the desire protection, while maintaining efficient operations across the network. Further, using existing protocol services could simplify the security protocol design, and thus facilitate the verification process. Security protocols which are compatible with conventional protocols would enjoy the additional benefits of modularity and evolvability.

The Inclusion Principle specifically addresses multilevel security concerns. This principle requires a separate suite of security protocols for each sensitivity level processed. The model prescribed by the Inclusion Principle vertically partitions the horizontal protocol layers, into compartments of different sensitivity levels. Essentially, the Inclusion Principle describes what should already be intuitively obvious--the higher the sensitivity level of information, the greater the protection needed to secure that information.

An advantage of the Inclusion Principle is that security protocols may be designed to provide the functionality and assurance appropriate to the level of information being processed. Thus, the cost of protection would compare favorably with the degree of security rendered. An additional benefit of information integrity is realized from separate security protocols for each sensitivity level, because processes communicating across a multilevel secure network will share equivalent security levels.

The Support Principle maintains that security protocols alone, cannot provide adequate network security. Appropriate hardware and software mechanisms must be provided to support the communications exchange and access controls, as specified by security protocols.

Hardware mechanisms are needed to protect primary and virtual memories as well as the state of the CPU during process execution. Software mechanisms augment the security provided by network hardware. Access control mechanisms enable network hosts to identify the current privileges of all executing processes, and encryption may be used to secure the communications path between network processes. Together, hardware and software mechanisms may be used to effectively isolate security functions from the possible misuse or infiltration by other network system or user processes. Thus, both hardware and software protection mechanisms are needed to provide adequate network security. They must enforce the rules of security protocols and implement network security services.

If we are ever going to bridge the gap between security and other aspects of network technology, we must adopt standard security protocols. Collectively, these design principles offer significant implications for such standards. In order to provide truly secure and reliable multilevel network communications, standard security protocols should incorporate the following design attributes: They should be

compatible with conventional protocols; they should include the capability to simultaneously protect multiple sensitivity levels; they should be properly supported by system's hardware and software protection mechanisms.

LIST OF REFERENCES

1. Fraim, L. J., "SCOMP: A Solution to the Multilevel Security Problem," Computer, pp. 26-34, July 1983.

2. Schell, R. R., Tao, T. F., and Heckman, M., "Designing the GEMSOS Security Kernel for Security and Performance," Proceedings of the 8th National Computer Security Conference, pp. 108-119, 1985.

3. U.S. Department of Defense, DOD 5200.28-STD, Department of Defense Trusted Computer System Evaluation Criteria, National Computer Security Center, December 1985.

4. National Computer Security Center, NCSC-TG-005, Trusted Network Interpretation of the Trusted Computer System Evaluation Criteria, National Computer Security Center, 31 July 1987.

5. International Organization for Standardization, Technical Committee 97, Addendum to ISO 7498 on Security Architecture, February 1985.

6. Pfleeger, C. P., Security in Computing, Prentice Hall, Inc., 1989.

7. Abrams, M. D., and Podell, H. J., Tutorial: Computer and Network Security, IEEE Computer Society Press, 1987.

8. Arsenault, A. W., "Developments in Guidance for Trusted Computer Networks," Proceedings of the 10th National Computer Security Conference, pp. 1-8, 1987.

9. Stallings, W., Tutorial: Computer Communications Architectures, Protocols, and Standards, IEEE Computer Society Press, 1985.

10. Stallings, W., Handbook of Computer-Communication Standards, Department of Defense (DOD) Protocol Standards, Howard W. Sams and Company, 1987.

11. Stallings, W., Handbook of Computer-Communications Standards, The Open Systems Interconnection (OSI) Model and OSI-Related Standards, Howard W. Sams and Company, 1987.

12. Folts, H. C., "A Tutorial on the Open Systems Interconnection Reference Model," Open Systems Data Transfer, pp. 2-21, June 1982.

13. International Organization for Standardization, Technical Committee 97, Addendum to ISO 7498 on Security Architecture, February 1985.

14. Brandstad, D. K., "Considerations for Security in the OSI Architecture," Proceedings of the 10th National Computer Security Conference, pp. 9-14, 1987.

15. Baker, P. C., et al., "A1 Assurance for Internet System: Doing the Job," Proceedings of the 9th National Computer Security Conference, pp. 130-137, 1986.

16. Millen, J. K., "A Network Security Perspective," Proceedings of the 9th National Computer Security Conference, pp. 8-15, 1986.

17. U. S. Department of Defense, OPNAVINST 5510.1H, National Security Manual, 1987.

18. Walker, S. T., "Network Security Overview," Proceedings of 1985 Symposium on Security and Privacy, pp. 62-76, 1985.

19. Hsiao, D. K., Kerr, D. S. and Madnick, S. E., Computer Security, Academic Press, 1979.

20. Silberschatz, A. and Peterson, James L., Operating Systems Concepts, Addison-Wesley Publishing Company, 1988.

21. Rogers, H. L., "An Overview of the Caneware Program," Proceedings of the 10th National Computer Security Conference, pp. 172-174, 1987.

22. Saydjari, O. S., Beckman, J. M. and Leaman, J. R., "Locking Computers Securely," Proceedings of the 10th National Computer Security Conference, pp. 129-141, 1987.

23. Denning, D. E., Cryptography and Data Security, Addison-Wesley Publishing Company, 1982.

24. Schnackenberg, D., "Applying the Orange Book to an MLS LAN," Proceedings of the 10th National Computer Security Conference, pp. 51-55, 1987.

25. Ames, S. R., Jr., Gasser, M. and Schell, R. R., "Security Kernel Design and Implementation: An Introduction," Computer, pp. 14-22, July 1983.

26. Popek, G. and Kline, C. "Encryption Protocols, Public Key Algorithms and Digital Signatures in Computer Networks," Foundations of Secure Computation, Academic Press, pp. 133-155, 1978.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943-5000	2
3.	Department Chairman, Code 54 Department of Administrative Sciences Naval Postgraduate School Monterey, California 93943-5000	1
4.	Curricular Officer, Code 37 Computer Technology Naval Postgraduate School Monterey, California 93943-5000	1
5.	Professor David K. Hsiao, Code 52Hq Department of Computer Science Naval Postgraduate School Monterey, California 93943-5000	2
6.	Assistant Professor Magdi N. Kamel, Code 54Ka Department of Administrative Sciences Naval Postgraduate School Monterey, California 93943-5000	2
7.	Lieutenant Claudia J. Kiefer 3273 Old Bridgeport Rd. San Diego, California 92111	2